AMENDMENT TO THE CLAIMS

This listing of claims will replace all prior versions, and listing, of claims in the application.

1. (Original) A method for use in developing a program, comprising compiling at least a portion of a source code program defined by a waypoint during the editing of the source code program.

2. (Original) The method of claim 1, wherein compiling includes:
identifying the waypoint in an edited source code during editing of the source code; and
compiling the source code up to the identified waypoint before completing the edit of the source code.

3. (Original) The method of claim 1, wherein identifying the waypoint includes one of identifying the waypoint from a static definition and identifying the waypoint from a dynamic definition.

4. (Original) The method claim 1, further comprising:
identifying a second waypoint in the source code during editing of the source code; and
compiling the source code from the first waypoint to the second waypoint before completing editing of the source code.

5. (Original) The method of claim 1, further comprising:
completing editing of the source code; and
compiling the source code from the second waypoint to the end of the source code.

6. (Original) The method of claim 1, further comprising saving the edited source code.

7. (Original) The method of claim 1, further comprising compiling the source code from the waypoint to the end of the source code upon completing editing of the source code.

8. (Original) A method for use in developing a program, comprising:

identifying a waypoint in an edited source code program during editing of the source code program; and

compiling the source code program up to the identified waypoint before completing editing of the source code program.

9. (Original) The method of claim 8, wherein identifying the waypoint includes one of identifying the waypoint from a static definition and identifying the waypoint from a dynamic definition.

10. (Original) The method claim 8, further comprising:

identifying a second waypoint in the edited source code program during editing of the source code program; and

compiling the source code program from the first waypoint to the second waypoint before completing editing of the source code program.

11. (Original) The method of claim 8, further comprising compiling the source code program from the waypoint to the end of the source code program upon completing editing of the source code program.

12. (Currently Amended) A method for modifying a compiler to engage in rapid compilation, comprising:

identifying a file reader portion of the compiler; and

modifying the identified file reader to read a portion of a source code program defined by a waypoint from a standard input ~~open~~.

13. (Original) The method of claim 12, wherein modifying the identified file reader to read from the standard input includes modifying the identified file reader to read from an open system call.

14. (Original) The method of claim 13, wherein modifying the identified file reader to read from the open system call includes modifying the identified file reader to read from a UNIX gcc command.

15. (Original) The method of claim 12, wherein the waypoint is identified by one of identifying the waypoint from a static definition and identifying the waypoint from a dynamic definition.

16. (Original) The method of claim 12, wherein the waypoint defines a lower bound of the portion of the source code program.

17. (Original) The method of claim 12, wherein the waypoint defines an upper bound of the portion of the source code program.

18. (Original) A method for suspending compiler execution prior to reaching the end of a source code program, comprising:

identifying a waypoint in the source code program;

compiling a portion of the source code program whose lower bound is defined by the identified waypoint; and

suspending compilation of the source code program once the portion whose lower bound is identified by the waypoint is compiled.

19. (Original) The method of claim 18, wherein the waypoint is identified by one of identifying the waypoint from a static definition and identifying the waypoint from a dynamic definition.

20. (Original) The method of claim 18, wherein suspending compilation of the source code program once the portion whose lower bound is identified by the waypoint is compiled includes at least one of removing a corresponding task from a work queue in an IDE, storing the compiled code in a shadow location, and suppressing errors or warning.

21. (Original) The method of claim 18, wherein the upper bound of the portion is defined by the start of the source code program or another waypoint.

22. (Original) A method for resuming compiler execution of a suspended compilation, comprising:

triggering the compilation of a portion of a source code program whose upper bound is defined by an identified waypoint; and

compiling the portion of the source code program whose upper bound is defined by the identified waypoint.

23. (Original) The method of claim 22, wherein triggering the compilation of the portion of the source code includes identifying the waypoint.

24. (Original) A method for identifying a command and associating it with a file that is being edited, comprising:

modifying a file reader of a compiler to read from a standard input; and

triggering the compilation of a portion of a source code program whose upper bound is defined by an identified waypoint;

invoking the compiler to read the file from the modified file reader through the standard input.

25. (Original) The method of claim 24, wherein modifying the file reader to read from the standard input includes modifying the identified file reader to read from an open system call.

26. (Original) The method of claim 24, wherein modifying the file reader to read from the open system call includes modifying the identified file reader to read from a UNIX gcc command.

27. (Original) The method of claim 24, wherein triggering the compilation of the portion of the source code includes identifying the waypoint.

28. (Original) A method for building a source code program capable of suspending and resuming compilation, comprising:

identifying a waypoint in a source code program being edited;

triggering a compilation of a portion of the source code program defined by the waypoint;

compiling the portion of the source code program defined by the waypoint;

suspending the compilation of the portion defined by the waypoint once the compilation reaches the waypoint;

triggering the compilation of the remainder of the source code program; and

resuming the compilation of the source code program to compile the remainder.

29. (Original) The method of claim 28, wherein the waypoint is identified by one of identifying the waypoint from a static definition and identifying the waypoint from a dynamic definition.

30. (Original) The method of claim 28, wherein triggering the compilation of the portion of the source code includes identifying the waypoint.

31. (Original) The method of claim 28, wherein suspending compilation of the source code program once the portion whose lower bound is identified by the waypoint is compiled includes at least one of removing a corresponding task from a work queue in an IDE, storing the compiled code in a shadow location, and suppressing errors or warning.

32. (Original) The method of claim 28, wherein the upper bound of the portion is defined by the start of the source code program or another waypoint.

33. (Original) The method of claim 28, wherein triggering the compilation of the remainder of the source code program includes identifying a second waypoint, saving the source code program, or ending an editing session.

34. (Original) A method for using a UNIX standard input read mechanism for speculative compilation of a source code program, comprising:

identifying a waypoint in an edited source code program during editing of the source code program; and

invoking a compile of at least a portion of a source code program defined by a waypoint during the editing of the source code program with a UNIX input read mechanism.

35. (Original) The method of claim 34, wherein the portion comprises a portion of the source code program defined by the start of the source code program and the waypoint.

36. (Original) The method of claim 34, wherein the portion comprises a portion of the source code program defined by the waypoint and the end of the source code program.

37. (Original) The method of claim 34, wherein the waypoint is identified by one of identifying the waypoint from a static definition and identifying the waypoint from a dynamic definition.

38. (Original) A method for managing the output of a compile, comprising:

compiling at least a portion of a source code program defined by a waypoint during the editing of the source code program in a first phase;

compiling the remainder of the source code program in a subsequent phase; and

notifying a user of any errors that may have occurred during the compilation.

39. (Original) The method of claim 38, wherein the portion comprises a portion of the source code program defined by the start of the source code program and the waypoint.

40. (Original) The method of claim 38, wherein the portion comprises a portion of the source code program defined by the waypoint and the end of the source code program.

41. (Original) The method of claim 38, wherein the waypoint is identified by one of identifying the waypoint from a static definition and identifying the waypoint from a dynamic definition.

42. (Original) The method of claim 38, further comprising scrapping the compiled first and second portions.

43. (Original) The method of claim 42, wherein scrapping the compiled first and second portions includes one of scrapping the compiled first and second portions responsive to the notification and scrapping the compiled first and second portions responsive to a user input.

44. (Original) A method for use in developing a program, comprising:

identifying at least two or more instructions in a file to compile; and
compiling the identified instructions while the file is being edited.

45. (Original) The method of claim 44, wherein the instructions are identified at a predetermined line number in the source code program, identifying the instructions at the point of insertion for a text editor, identifying the instructions after a predetermined number of branches as conditionals, identifying the instructions at a predetermined text offset.

46. (Original) The method claim 44, further comprising:
identifying at least two more instructions in the file during editing; and
compiling the second two or more instruction while the file is being edited.

47. (Original) The method of claim 44, further comprising:
completing editing of the file; and
compiling the remainder of the edited file.

48. (Original) The method of claim 44, further comprising saving the edited file.

49. (Original) The method of claim 44, further comprising compiling the remainder of the edited file upon completing editing of the file.

50. (Original) A method for compiling a source code program, comprising:
identifying an upper bound for a portion of the source code program to compile;
identifying a lower bound for the portion; and
compiling the portion defined by the upper and lower bounds during an editing session on the source code program.

51. (Original) The method of claim 50, wherein at least one of identifying the upper bound and identifying the lower bound includes one of identifying the bound from a static definition and identifying the bound from a dynamic definition.

52. (Original) The method claim 50, further comprising:

identifying a third bound in the edited source code during editing of the source code; and

compiling the source code from the lower bound to the third bound before completing editing of the source code.

53. (Original) The method of claim 50, further comprising compiling the source code from the lower bound to the end of the source code upon completing editing of the source code.